AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

1. (Previously Presented) A method of using a virtual machine monitor and an operating system on computer hardware in a computer, the method comprising:

interposing the virtual machine monitor between the computer hardware and the operating system at runtime, wherein the interposing occurs after booting of the computer, and wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the computer hardware; and

booting the operating system on the computer hardware before interposing the virtual machine monitor at runtime.

2. (Cancelled)

3. (Previously Presented) The method of claim 1, further comprising booting the virtual machine monitor on the computer hardware, booting the operating system on the virtual machine monitor, and devirtualizing the computer hardware before interposing the virtual machine monitor at runtime.

4. (Previously Presented) The method of claim 1, further comprising devirtualizing the computer hardware at runtime after the virtual machine monitor has been interposed.

5. (Original) The method of claim 1, wherein the computer hardware includes a CPU; and wherein the virtual machine monitor is interposed on the CPU.

6. (Currently Amended) The method of claim 5, wherein the computer hardware further includes memory, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein interposing the virtual machine monitor on the CPU includes:

causing privileged instructions to trap to the virtual machine monitor, and

redirecting interrupts to the corresponding virtual machine monitor CPU interrupt handlers instead of to the operating system CPU interrupt handlers.

7. (Original) The method of claim 6, wherein the privileged instructions are caused to trap to the virtual machine monitor by causing the operating system to run at a reduced privilege level; and wherein interposing the virtual machine monitor on the CPU further includes returning control to the operating system at the reduced privilege level.

8. (Previously Presented) The method of claim 6, wherein the privileged instructions are caused to trap to the virtual machine monitor by using a kernel module of the operating system to reduce a privilege level of the operating system from a higher privilege level.

9. (Previously Presented) The method of claim 6, wherein interposing the virtual machine monitor on the CPU further includes disabling physical memory access by the operating system.

10. ((Previously Presented) The method of claim 6, wherein interposing the virtual machine monitor on the CPU further includes loading the virtual machine monitor into the memory.

11. (Previously Presented) The method of claim 10, further comprising using a kernel module of the operating system to allocate memory within the operating system, pin the allocated memory, and load the virtual machine monitor into the pinned memory.

12. (Original) The method of claim 5, wherein the computer hardware includes memory; and wherein the virtual machine monitor is also interposed on the memory.

13. (Previously Presented) The method of claim 12, wherein interposing the virtual machine monitor on the memory includes partitioning the memory to provide partitions, and giving the virtual machine monitor access to at least one of the partitions.

14. (Original) The method of claim 12, wherein interposing the virtual machine monitor on the memory includes using a kernel module of the operating system to allocate a block of the memory, pin the block to prevent the operating system from using the block, and allocate the pinned block to the virtual machine monitor.

15. (Previously Presented) The method of claim 12, wherein interposing the virtual machine monitor on the memory includes commencing using the virtual machine monitor at runtime to manage memory translation.

16. (Original) The method of claim 5, wherein the computer hardware includes an I/O device, and wherein the virtual machine monitor is also interposed on the I/O device.

17. (Previously Presented) The method of claim 16, wherein the operating system includes a dual-mode driver that performs direct hardware control in a first mode and communicates with a device driver of the virtual machine monitor in a second mode; and wherein interposing the virtual machine monitor on the I/O device includes:

setting the dual-mode driver to the second mode; and

redirecting I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system.

18. (Previously Presented) The method of claim 16, wherein interposing the virtual machine monitor on the I/O device includes commencing I/O emulation of the I/O device at runtime.

19. (Previously Presented) A method of using a virtual machine monitor and an operating system on virtualized computer hardware, the method comprising devirtualizing the virtualized computer hardware at runtime of a computer containing the virtualized computer hardware, wherein runtime includes a period of execution in the computer after booting and before shutdown,

wherein devirtualizing the virtualized computer hardware comprises stopping the virtual machine monitor.

20. (Previously Presented) The method of claim 19, wherein the virtualized computer hardware includes a CPU; and wherein the CPU is devirtualized at runtime.

21. (Currently Amended) The method of claim 20, wherein the virtualized computer hardware further includes physical memory, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein devirtualizing the CPU includes redirecting interrupts to the corresponding operating system CPU interrupt handlers instead of to the virtual machine monitor CPU interrupt handlers.

22. (Previously Presented) The method of claim 21, wherein devirtualizing the CPU further includes restoring a privilege level of the operating system from a less privileged mode to a more privileged mode.

23. (Previously Presented) The method of claim 21, wherein devirtualizing the CPU further includes enabling physical memory access by the operating system.

24. (Previously Presented) The method of claim 21, wherein devirtualizing the CPU further includes unloading the virtual machine monitor from the physical memory.

25. (Previously Presented) The method of claim 19, wherein the virtualized computer hardware includes memory; and wherein the memory is devirtualized at runtime.

26. (Original) The method of claim 25, wherein memory was allocated from the operating system to the virtual machine monitor during virtualization of the memory; and wherein devirtualizing the memory includes returning the allocated memory to the operating system.

27. (Previously Presented) The method of claim 25, wherein devirtualizing the memory includes remapping physical memory and using the operating system to manage address translation with respect to the devirtualized memory.

28. (Previously Presented) The method of claim 19, wherein the virtualized computer hardware includes an I/O device, and wherein the I/O device is devirtualized at runtime.

29. (Currently Amended) The method of claim 28, wherein the operating system includes a dual-mode driver that performs direct hardware control in a first mode and communicates with a device driver of the virtual machine monitor in a second mode; and wherein devirtualizing the I/O device includes:

setting the dual-mode driver to the first mode from the second mode, and

redirecting I/O interrupts to handlers in the operating system instead of [[the]]handlers in the virtual machine monitor.

30. (Original) The method of claim 28, wherein devirtualizing the I/O device includes ceasing emulation of the I/O device at runtime.

31. (Previously Presented) A computer comprising hardware, the hardware including memory, the memory encoded with an operating system, a virtual machine monitor, and code for interposing the virtual machine monitor between the operating system and the hardware at runtime, wherein the interposing occurs after booting of the computer,

wherein the operating system is to be booted in the computer before interposing the virtual machine monitor.

32. (Currently Amended) The computer of claim 31, wherein the hardware further includes a CPU, wherein the virtual machine monitor is interposed on the CPU at runtime, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein the interposing code is to cause privileged instructions to trap to the virtual machine monitor, and to redirect interrupts and traps to the corresponding virtual machine monitor CPU interrupt handlers instead of to the operating system CPU interrupt handlers.

33. (Previously Presented) The computer of claim 32, wherein the interposing code is to cause privileged instructions to trap to the virtual machine monitor by causing the operating system to run at a reduced privilege level from a higher privilege level; and wherein the interposing code is to reduce a privilege level of the operating system after redirecting the interrupts, and to return control to the operating system at the reduced privilege level.

34. (Previously Presented) The computer of claim 32, wherein the interposing code includes a kernel module of the operating system for reducing a privilege level of the operating system from a higher privilege level, whereby the privileged instructions trap to the virtual machine monitor.

35. (Previously Presented) The computer of claim 32, wherein the interposing code is to disable physical memory access by the operating system.

36. (Previously Presented) The computer of claim 31, wherein the interposing code includes a kernel module of the operating system for allocating a block of the memory, pinning the block to prevent the operating system from using the block, and allocating the pinned block to the virtual machine monitor, whereby the virtual machine monitor is interposed on the memory at runtime.

37. (Previously Presented) The computer claim 31, wherein the interposing code is to commence using the virtual machine monitor at runtime to manage memory translation, whereby the virtual machine monitor is interposed on the memory at runtime.

38. (Previously Presented) The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the interposing code includes an operating system dual-mode driver to perform direct hardware control in a first mode and to communicate with a device driver of the virtual machine monitor in a second mode; and wherein the interposing code is to set the dual-mode driver to the second mode, and to direct I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system, whereby the virtual machine monitor is interposed on the I/O device at runtime.

39. (Previously Presented) The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the operating system includes a dual-mode driver to perform direct hardware control in a first mode and to communicate with a device driver of the virtual machine monitor in a second mode; and wherein the interposing code is to set the dual-mode driver to the second mode, and to redirect I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system, whereby the virtual machine monitor is interposed on the I/O device.

40. (Previously Presented) The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the interposing code is to commence I/O emulation of the I/O device at runtime, whereby the virtual machine monitor is interposed on the I/O device at runtime.

41. (Previously Presented) A computer comprising hardware, the hardware including memory, the memory encoded with a virtual machine monitor to virtualize the hardware, and code for devirtualizing the hardware at runtime, wherein runtime includes a period of execution in the computer after booting and before shutdown, and wherein devirtualizing the hardware comprises stopping the virtual machine monitor.

42. (Previously Presented) The computer of claim 41, wherein the hardware further includes a CPU; and wherein the devirtualizing code is to devirtualize the CPU at runtime.

43. (Previously Presented) The computer of claim 42, wherein the memory is further encoded with an operating system including interrupt handlers; wherein the virtual machine monitor

includes interrupt handlers; and wherein the devirtualizing code is to redirect interrupts to the corresponding interrupt handlers of the operating system instead of to the interrupt handlers of the virtual machine monitor.

44. (Previously Presented) The computer of claim 43, wherein the devirtualizing code is to restore privilege level of the operating system from a lower privilege level to a higher privilege level.

45. (Previously Presented) The computer of claim 43, wherein the devirtualizing code is to enable physical memory access by the operating system.

46. (Previously Presented) The computer of claim 41, wherein the devirtualizing code is to devirtualize the memory at runtime.

47. (Previously Presented) The computer of claim 46, wherein the virtual machine monitor is to allocate memory from an operating system to the virtual machine monitor; and wherein the devirtualizing code is to return the allocated memory to the operating system.

48. (Cancelled)

49. (Previously Presented) The computer of claim 41, wherein the hardware includes an I/O device, wherein the virtual machine monitor is to virtualize the I/O device; and wherein the devirtualizing code is to devirtualize the I/O device at runtime.

50. (Previously Presented) The computer of claim 49, wherein the memory is further encoded with an operating system including dual-mode drivers to perform direct hardware control in a first mode and communicate with device drivers of the virtual machine monitor in a second mode; and wherein the devirtualizing code is to set the dual-mode drivers to the first mode from the second mode, and to redirect I/O interrupts to handlers in the operating system instead of to handlers in the virtual machine monitor.

51. (Previously Presented) The computer of claim 49, wherein the devirtualizing code is to cease emulation of the I/O device at runtime.

52. (Previously Presented) An article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by the computer causes the computer to:

virtualize at least a portion of the computer hardware at runtime by providing a virtual machine monitor between the operating system and the computer hardware, wherein the virtualizing occurs after booting of the computer and loading of the operating system, and

wherein the operating system is to be booted in the computer before virtualizing the at least a portion of the computer hardware at runtime.

53. (Previously Presented) The article of claim 52, wherein the computer hardware further includes a CPU, and wherein the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein the software is executable to cause privileged instructions to trap to the virtual machine monitor, and to cause interrupts and traps to be redirected to the corresponding virtual machine monitor interrupt handlers instead of to the operating system interrupt handlers.

54. (Previously Presented) The article of claim 53, wherein the software is executable to cause the privileged instructions to trap to the virtual machine monitor by reducing a privilege level of the operating system from a higher privilege level, and wherein the software causes control to be returned to the operating system at the reduced privilege level.

55. (Previously Presented) The article of claim 53, wherein the software is executable to cause physical memory access by the operating system to be disabled.

56. (Previously Presented) The article of claim 52, wherein the computer hardware includes memory, and wherein the virtual machine monitor is for causing a kernel module of the operating system to allocate a block of the memory, pin the block to prevent the operating system from using the block, and allocate the pinned block to the virtual machine monitor.

57. (Cancelled)

58. (Currently Amended) The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the software includes an operating system dual-mode driver to perform direct hardware control in a first mode and communicate with a corresponding device driver of a virtual machine monitor in a second mode; and wherein the dual-mode driver is set to the second mode when the at least the portion of the computer hardware is virtualized, and wherein I/O interrupts are redirected to interrupt handlers in the virtual machine monitor instead of [[the]]interrupt handlers in the operating system.

59. (Previously Presented) The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the operating system includes a dual-mode driver to perform direct hardware control in a first mode and communicate with a device driver of the virtual machine monitor in a second mode; and wherein the dual-mode driver is set to the second mode when the at least the portion of the computer hardware is virtualized, and wherein I/O interrupts are redirected from interrupt handlers in the operating system to interrupt handlers in the virtual machine monitor.

60. (Previously Presented) The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the software is executable to cause I/O emulation of the I/O device to commence at runtime.

61. (Original) An article for running an operating system and a virtual machine monitor on a computer, the computer including an I/O device, the article comprising computer memory encoded with an I/O driver having first and second modes of operation, the I/O driver operable in the first mode to interface directly between the operating system and the I/O device, the I/O driver operable in the second mode to interface between the operating system and a corresponding I/O driver of the virtual machine monitor.

62. (Previously Presented) An article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by a computer causes the computer to devirtualize at least a portion of virtualized hardware at runtime, wherein runtime is a period of execution in the computer after booting and before shutdown, and wherein devirtualizing the at least a portion of the virtualized hardware comprises stopping a virtual machine monitor interposed between the operating system and the hardware.

63. (Previously Presented) The article of claim 62, wherein the virtualized hardware includes a CPU; and wherein the software causes the CPU to be devirtualized at runtime.

64. (Previously Presented) The article of claim 63, wherein the virtualized hardware further includes memory, and wherein the memory is further encoded with the operating system including first interrupt handlers; wherein the software includes second interrupt handlers; and wherein the software is executable to cause interrupts to be redirected to the corresponding first interrupt handlers instead of to the second interrupt handlers.

65. (Previously Presented) The article of claim 64, wherein the software is executable to cause a privilege level of the operating system to be restored from a lower privilege level to a higher privilege level.

66. (Previously Presented) The article of claim 64, wherein the software is executable to cause physical memory access by the operating system to be enabled.

67. (Previously Presented) The article of claim 62, wherein the virtualized hardware includes a memory, and wherein the software is executable to cause the memory to be devirtualized at runtime.

68. (Currently Amended) The article of claim 67, wherein if a part of the memory was allocated from an operating system to the virtual machine monitor prior to the runtime devirtualization, the software is executable to cause ~~causes~~ the allocated memory to be returned to the operating system as part of the runtime devirtualization.

69. (Previously Presented) The article of claim 67, wherein the software is executable to cause physical memory to be remapped and wherein the software allows an operating system to manage address translation with respect to the devirtualized memory.

70. (Previously Presented) The article of claim 62, wherein the virtualized hardware includes an I/O device; and wherein the software is executable to cause the I/O device to be devirtualized at runtime.

71. (Previously Presented) The article of claim 70, wherein the virtualized hardware further includes a memory, and wherein the memory is further encoded with the operating system including dual-mode drivers that perform direct hardware control in a first mode and communicate with virtual device drivers in a second mode; and wherein the software is executable to cause the dual-mode drivers to be set to the first mode.

72. (Previously Presented) The article of claim 70, wherein the software is executable to cause emulation of the I/O device to cease at runtime.

73. (Previously Presented) The computer of claim 31, wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the

hardware such that the operating system no longer has direct control of the at least a portion of the hardware.

74. (Previously Presented) The article of claim 52, wherein providing the virtual machine monitor between the operating system and the computer hardware gives the virtual machine monitor direct control of at least a portion of the hardware such that the operating system no longer has direct control of the at least a portion of the hardware.